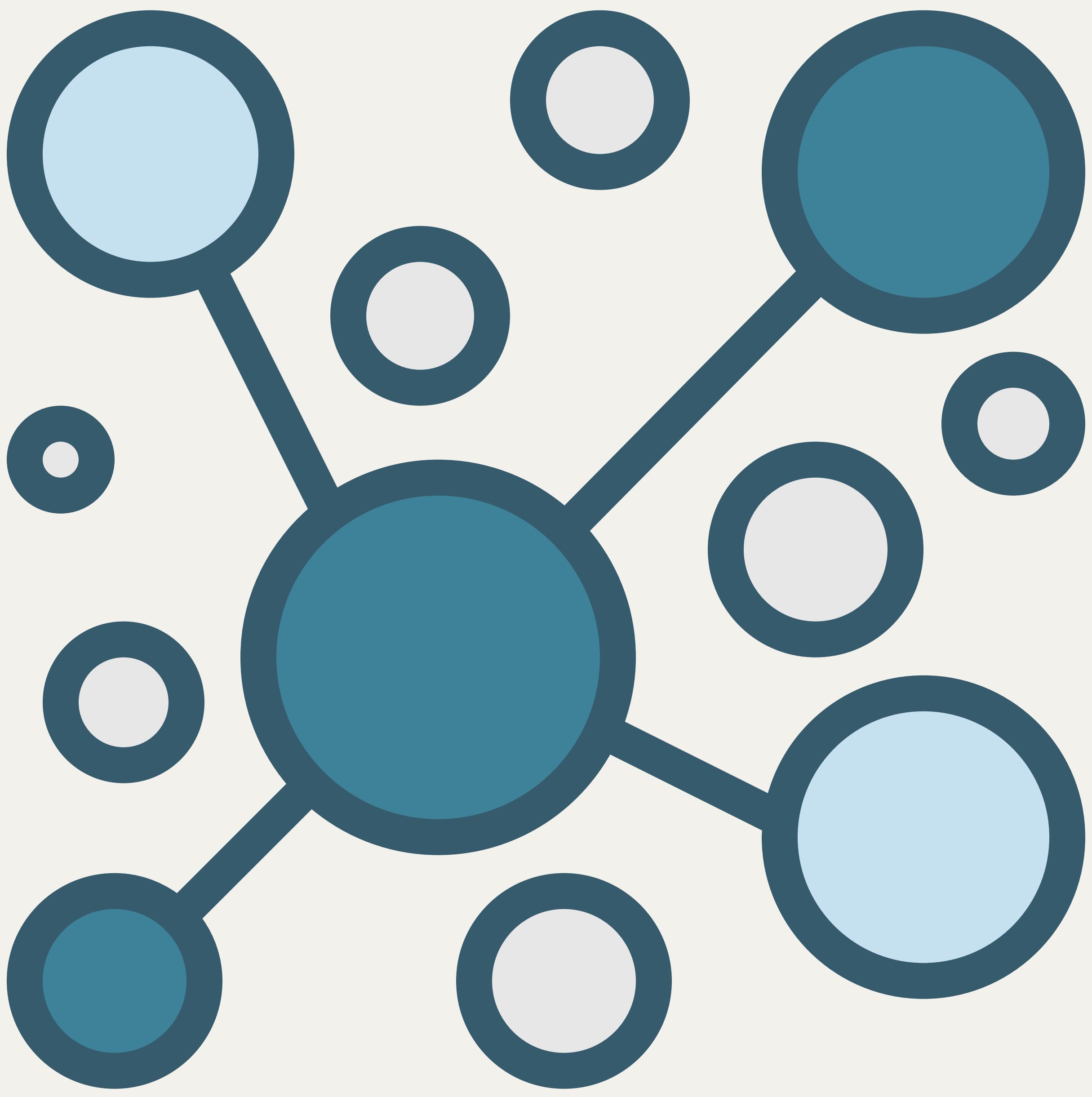
https://www.linkedin.com/company/code2career-ai

K-Means Clustering

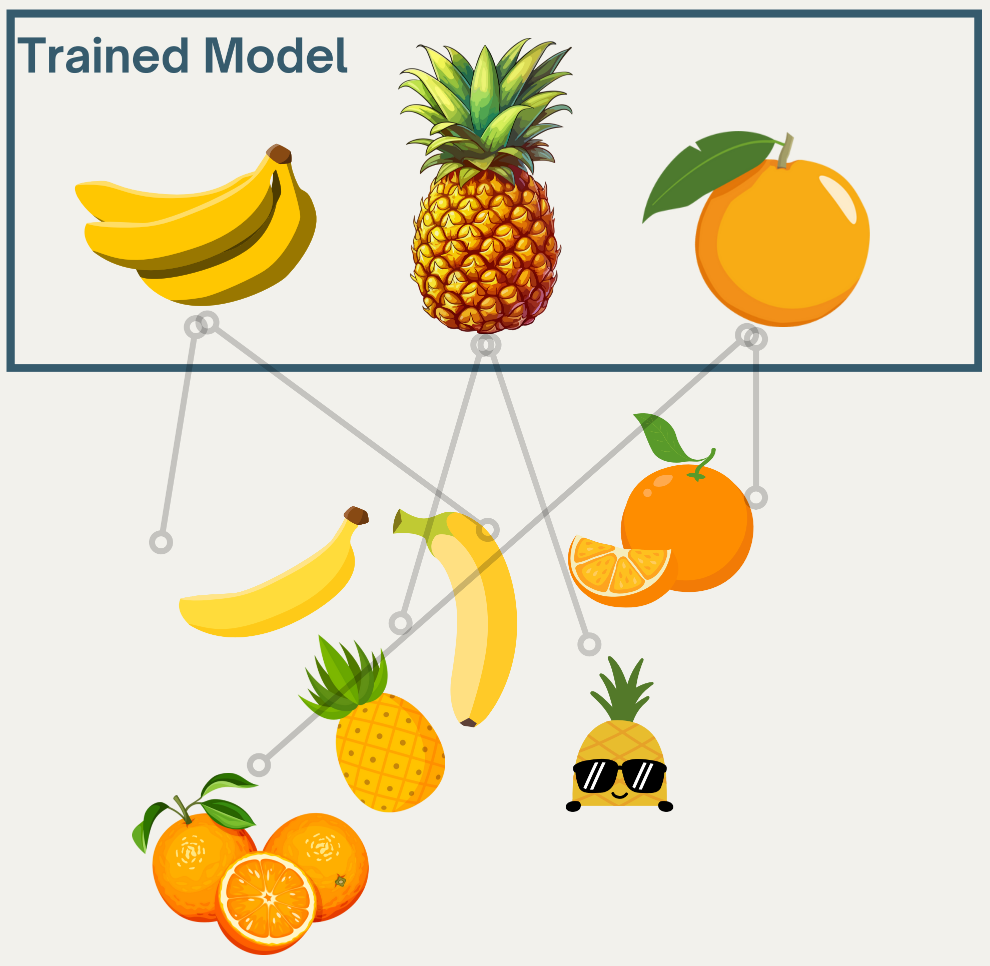
Clearly Explained



What is Unsupervised Learning?

Before we jump into what K-Means Clustering is, let see what the difference between **Unsupervised** and **Supervised** Learning is:

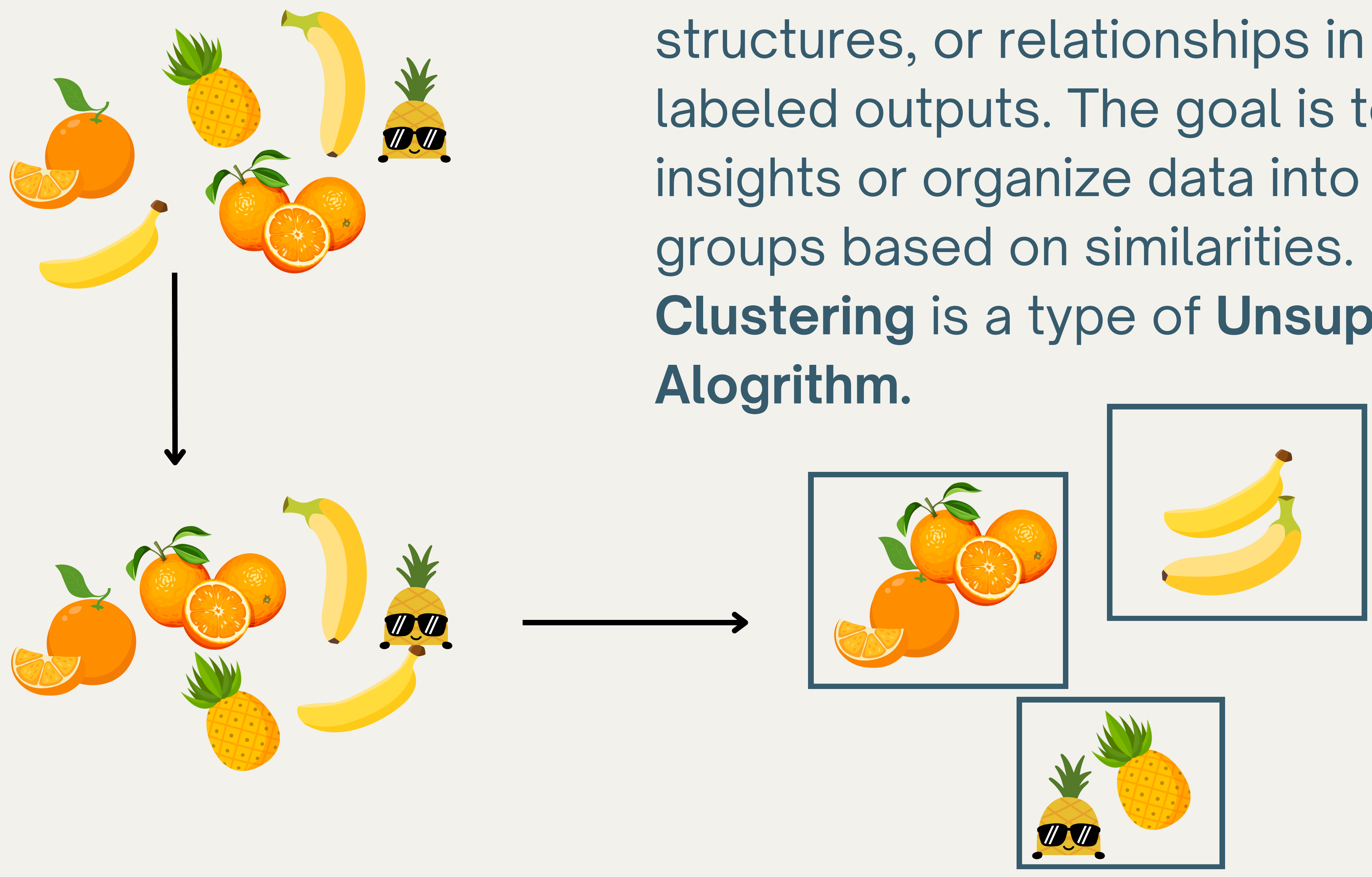
Supervised Learning



Supervised Learning usually consists of an algorithm that learns from labeled data. The model learns to recognize characteristics based on the training data and the label (Eg. Predicting Property prices based on size, location etc). It then is able to predict the label of unlabelled data on based on those characterics.

Unsupervised Learning

Unsupervised Learning is a type of machine learning where an algorithm learns patterns, structures, or relationships in data without any labeled outputs. The goal is to uncover hidden insights or organize data into meaningful groups based on similarities. **K-Means Clustering** is a type of **Unsupervised Learning Alogrithm.**

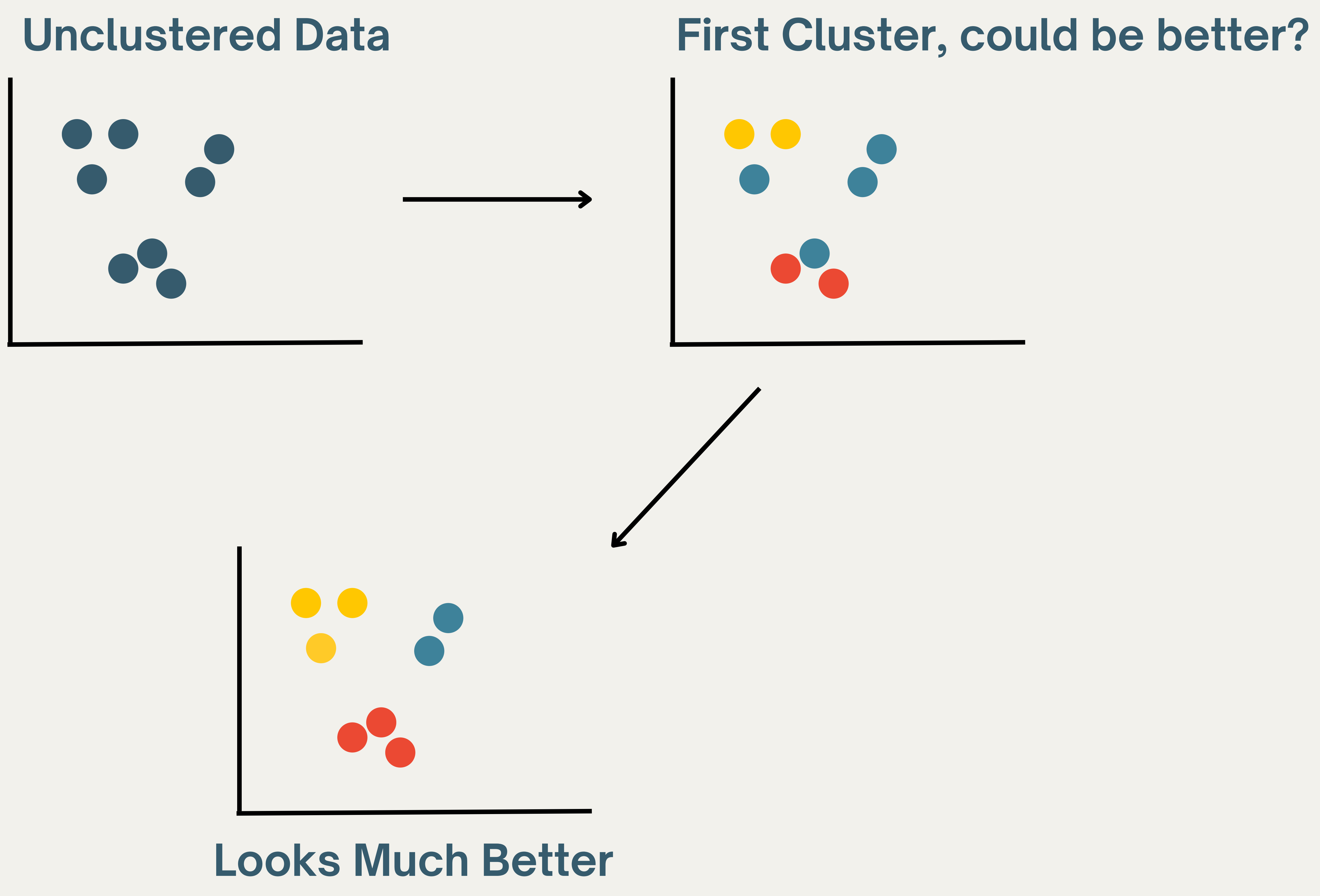


https://www.linkedin.com/company/code2career-ai

What is K-Means Clustering

As we mentioned in the previous page, **K-Means Clustering** is a Unsupervised Learning method. It works very simply:

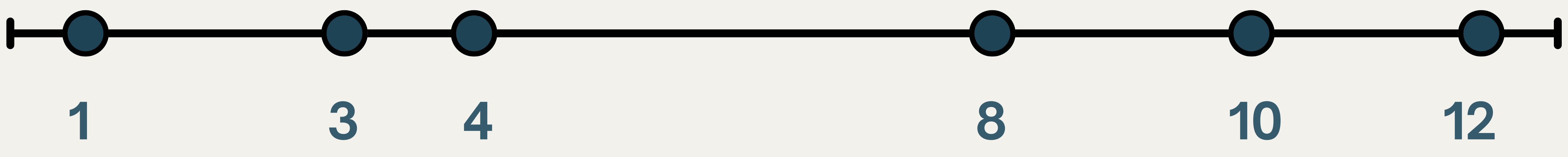
- Divides data into K distinct clusters
- Groups similar data points based on their features
- Ensures data points in a cluster are closer to each other than to points in other clusters



Since it's an unsupervised method, we don't have any labels to train a model on a training set. The K-means algorithm iterates by creating clusters (we give the number of clusters) and then measuring if it's good. So how does it create clusters and how does it measure the "goodness" of a cluster? Let's see how it works!

How does it work?

Let’s start with a one-dimensional data set : [1, 3, 4, 8, 10, 12]



- **Step 1: Decide on the number of clusters (K)**

We’ll create 2 clusters here. So we take K = 2.

- **Step 2: Decide on the number of clusters (K)**

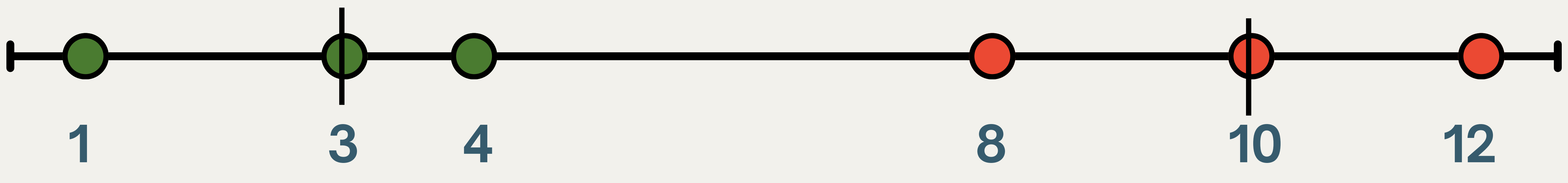
Randomly select two initial “**Centroids**” (equal to k) from the data points. Here centroid simply means the mean of all the points in a cluster. Since we don’t have a cluster yet, we will select 2 data points. The initial Centroids we picked here are : **[3, 10]**

- **Step 3: Calculate the Distance of Each Centroid from the data points**

Data Point	**Distance to Centroid 3**	**Distance to Centroid 10**	**Assigned Cluster**
1	2	9	3
3	0	7	3
4	1	6	3
8	5	2	10
10	7	0	10
12	9	2	10

How does it work?

We calculated the distance in the table previously. Since it’s one dimensional data, the distance is just the difference between the centroid and the data point. We assigned **each point to the cluster with closest centroid**



- **Step 4: Update the Centroids**

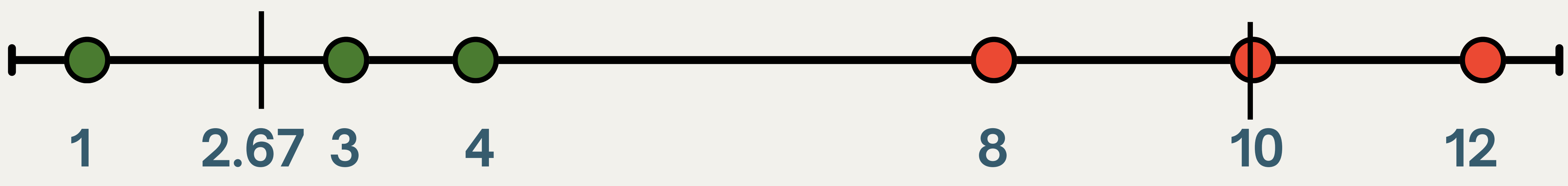
Now that we have our two clusters, we calculate new Centroids. It’s simply the mean of all the points in the cluster.

$$\textbf{New Centroid for Cluster 1} = \frac{1+3+4}{3} = \textbf{2.67}$$

$$\textbf{New Centroid for Cluster 2} = \frac{8+10+12}{3} = \textbf{10}$$

These two points are now our new Centroids **[2.67, 10]**

We then repeat steps 3 and 4 again. We will c



How does it work?

- **Step 5: Repeat Steps 3 - 4**

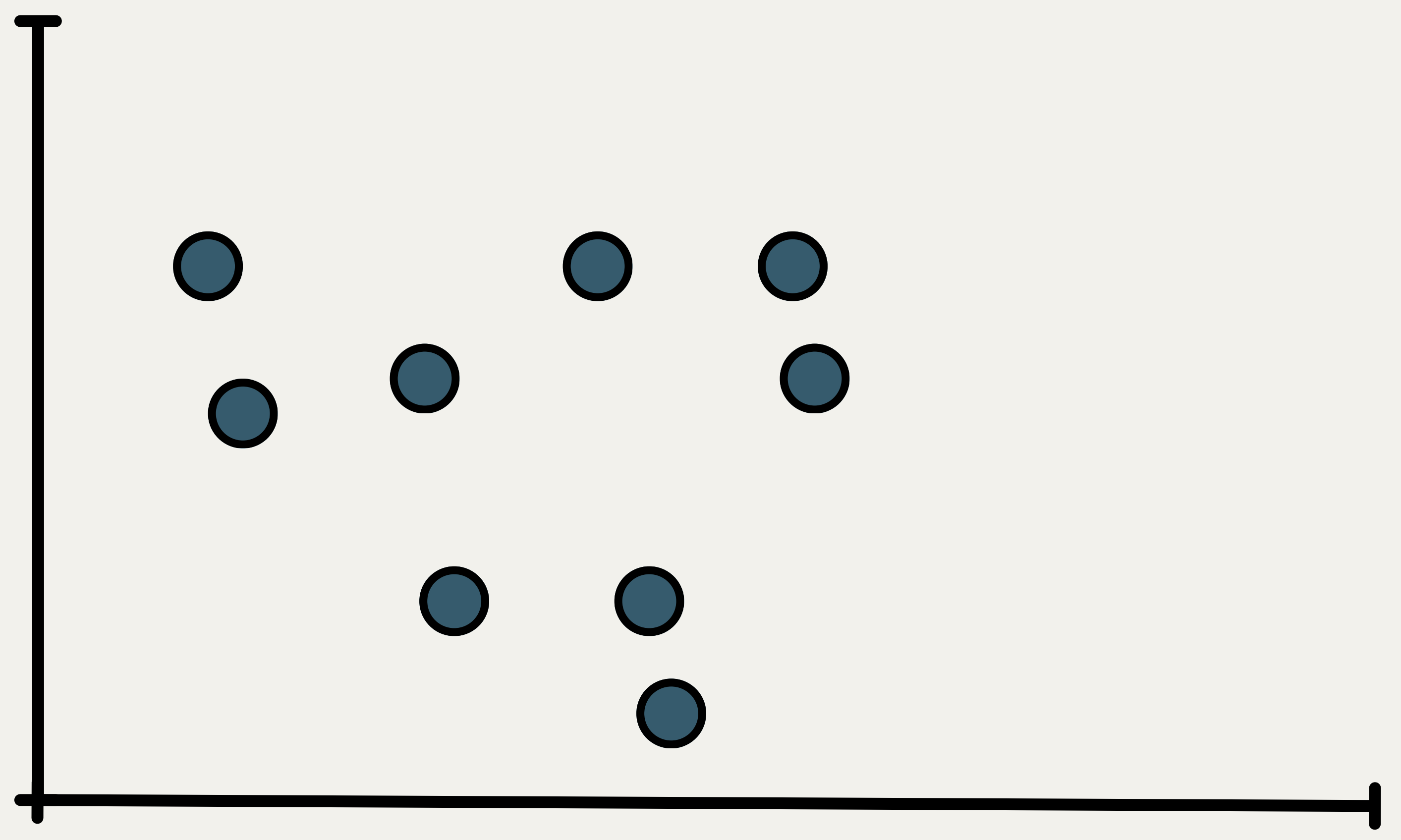
Data Point	Distance to Centroid 2.67	Distance to Centroid 10	Assigned Cluster
1	1.67	9	2.67
3	0.33	7	2.67
4	1.33	6	2.67
8	5.33	2	10
10	7.33	0	10
12	9.33	2	10

Now from what we can see above:

- Even though the Centroids changed (atleast one did), the clusters remained the same **[1,3,4]** & **[8,10,12]**.
- If we were to repeat steps 3 and 4 again, we would get the same result since the we'd get the same Centroids again. The alorithm thus stops here and we have the best presumed cluster.
- The alogrith stops when the Centroids don't change or when it reaches the max number of iterations.

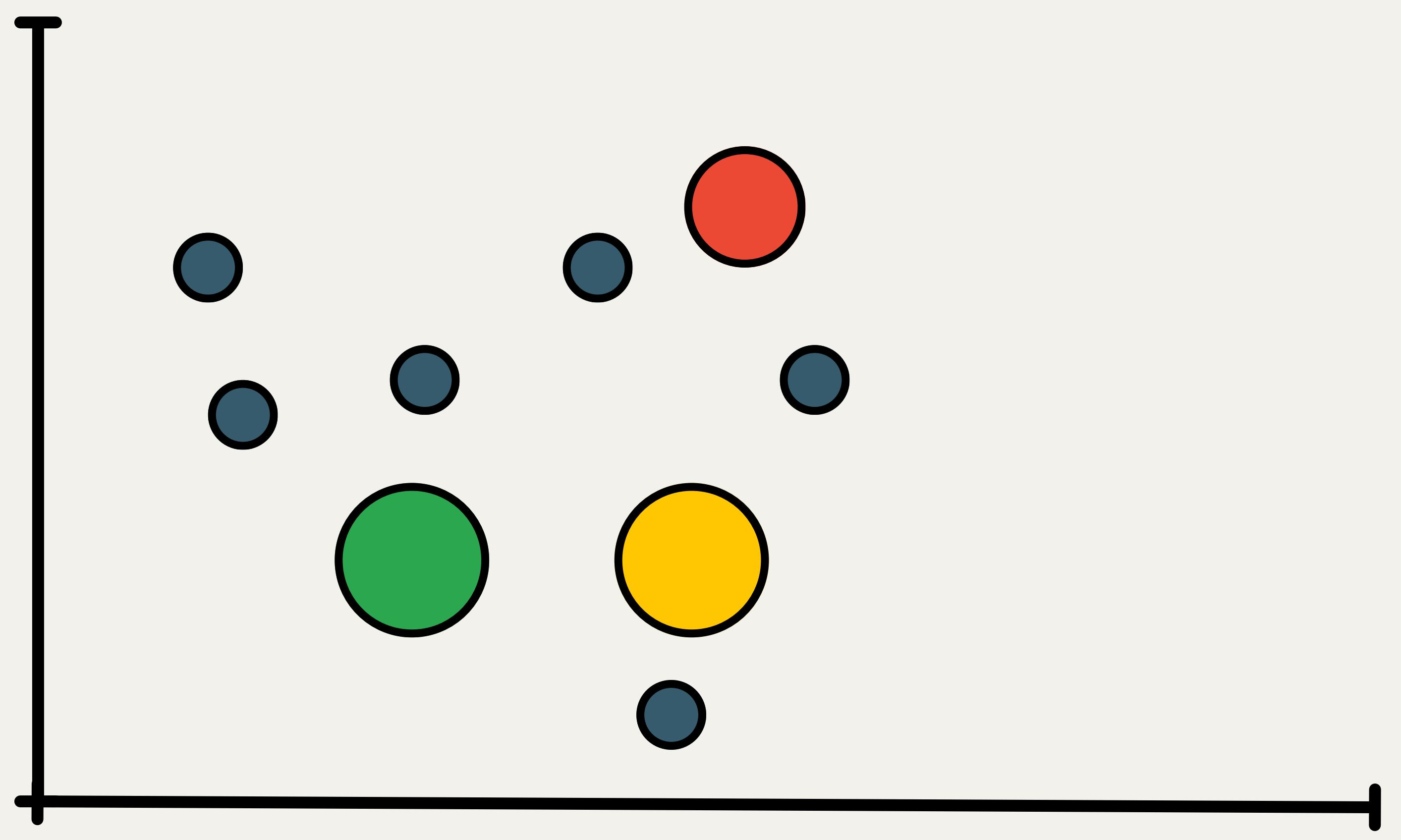
Now let's see how it would perform on a 2 Dimensional Dataset!

How does it work?



Step 1: Let’s create 3 clusters this time. So K = 3

Step 2: With a 2-D data set, we will again start with picking the Centroids. These will be 3 points randomly selected.



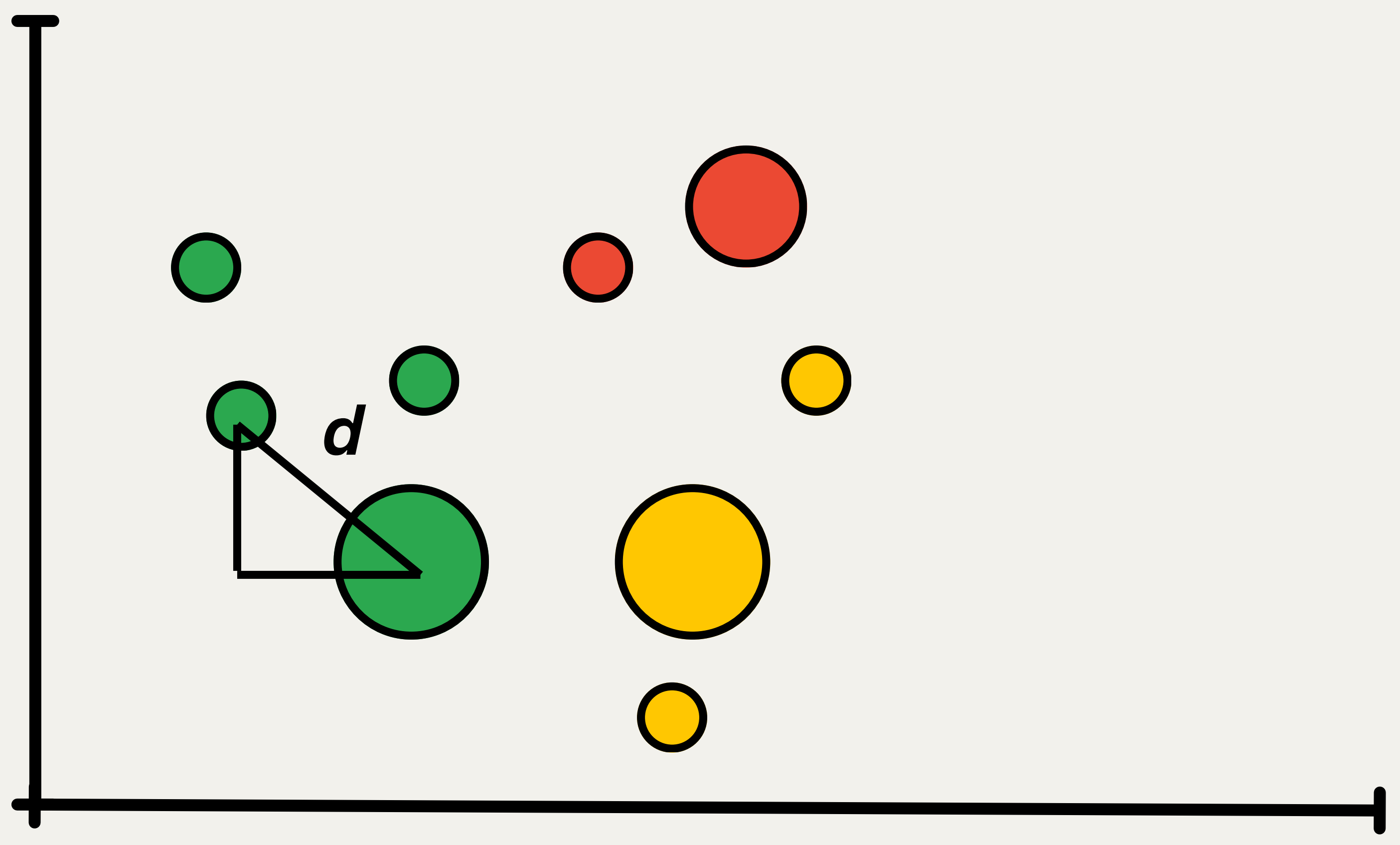
The **Red**, **Green** and **Yellow** points are our initial centroids.

How does it work?

Step 3: Similar to when we had 1-D data, we have to calculate the **distance of each point** from the centroids and then put them in the **cluster with the closest centroid.** Now, to calculate the distance in 2-D and more dimensions, we use something called Euclidean Distance ***d***

$$d = \sqrt[2]{(x_1 - x_2)^2 + (y_2 - y_1)^2}$$

Step 4: After calculating the distances, we would then assign each point to the cluster belonging to the Centroid that is at the least distance from it.



These are our new clusters. We now **update the centroids** again, by **calculating the mean of all the points in a particular cluster.**

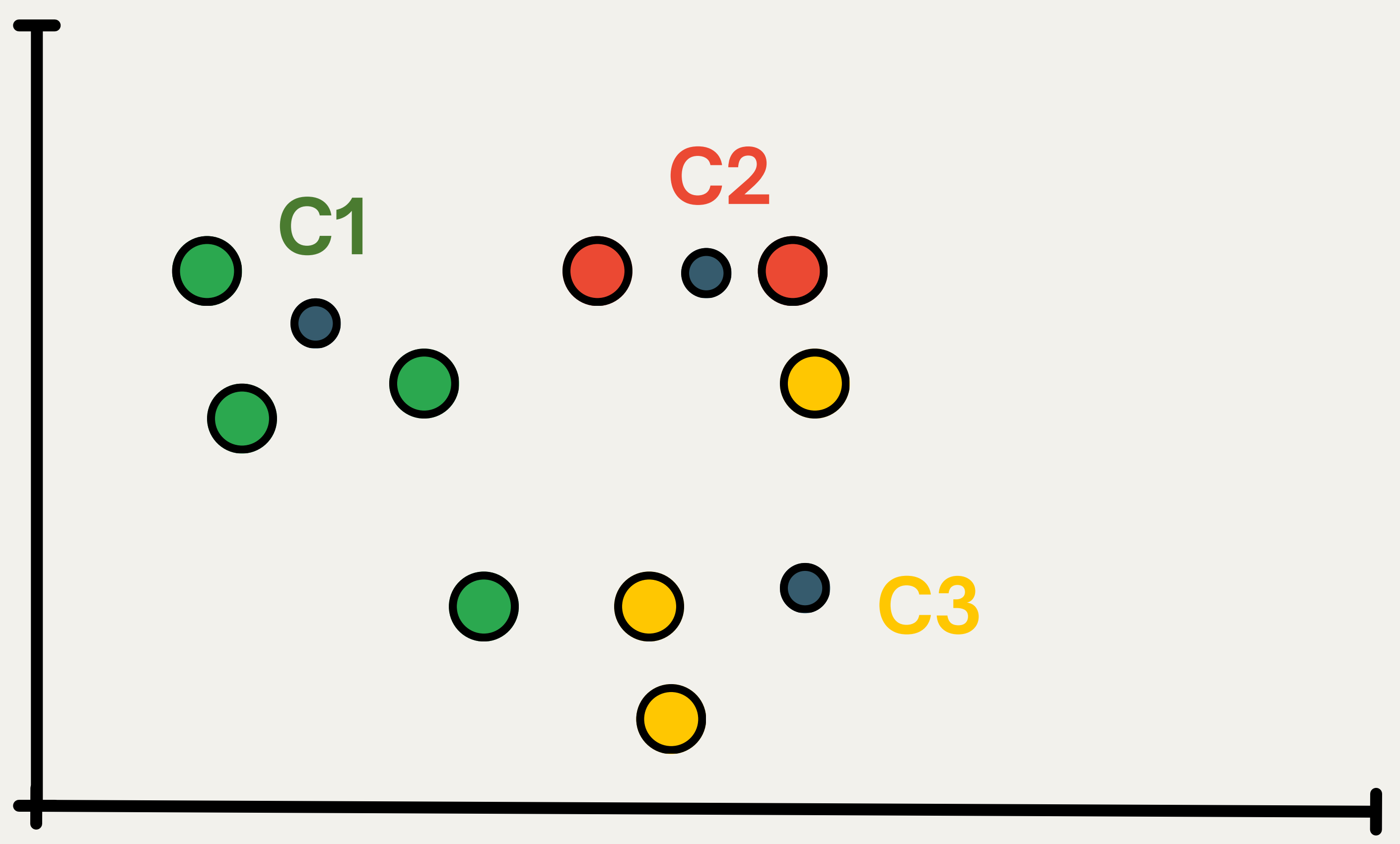
New Centroid for Cluster 1 = $\frac{x1 + x2 + x3 + x4}{4}, \frac{y1 + y2 + y3 + y4}{4}$

We do the same for **Cluster 2** and **Cluster 3.**

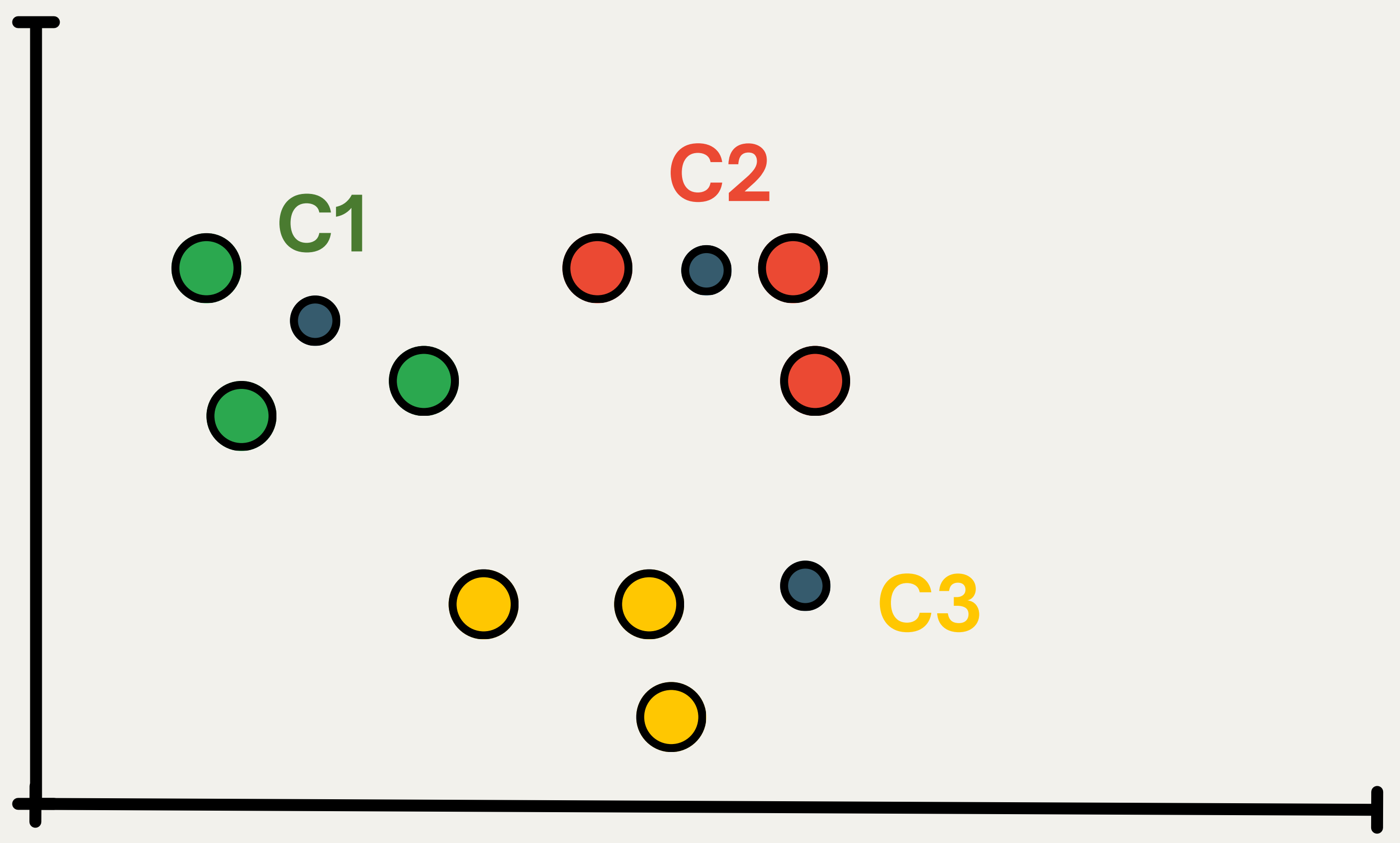
https://www.linkedin.com/company/code2career-ai

How does it work?

Step 5: Repeat Step 3 & 4



With our new Centroids **C1**, **C2** and **C3** we again calculate the distance of each point from each Centroid and bucket them in the cluster belonging to the nearest Centroid.



This looks like a much better set of clusters! The algorithm will continue to create new centroids and changing clusters till either the centroids don't change significantly or the max iterations is reached.

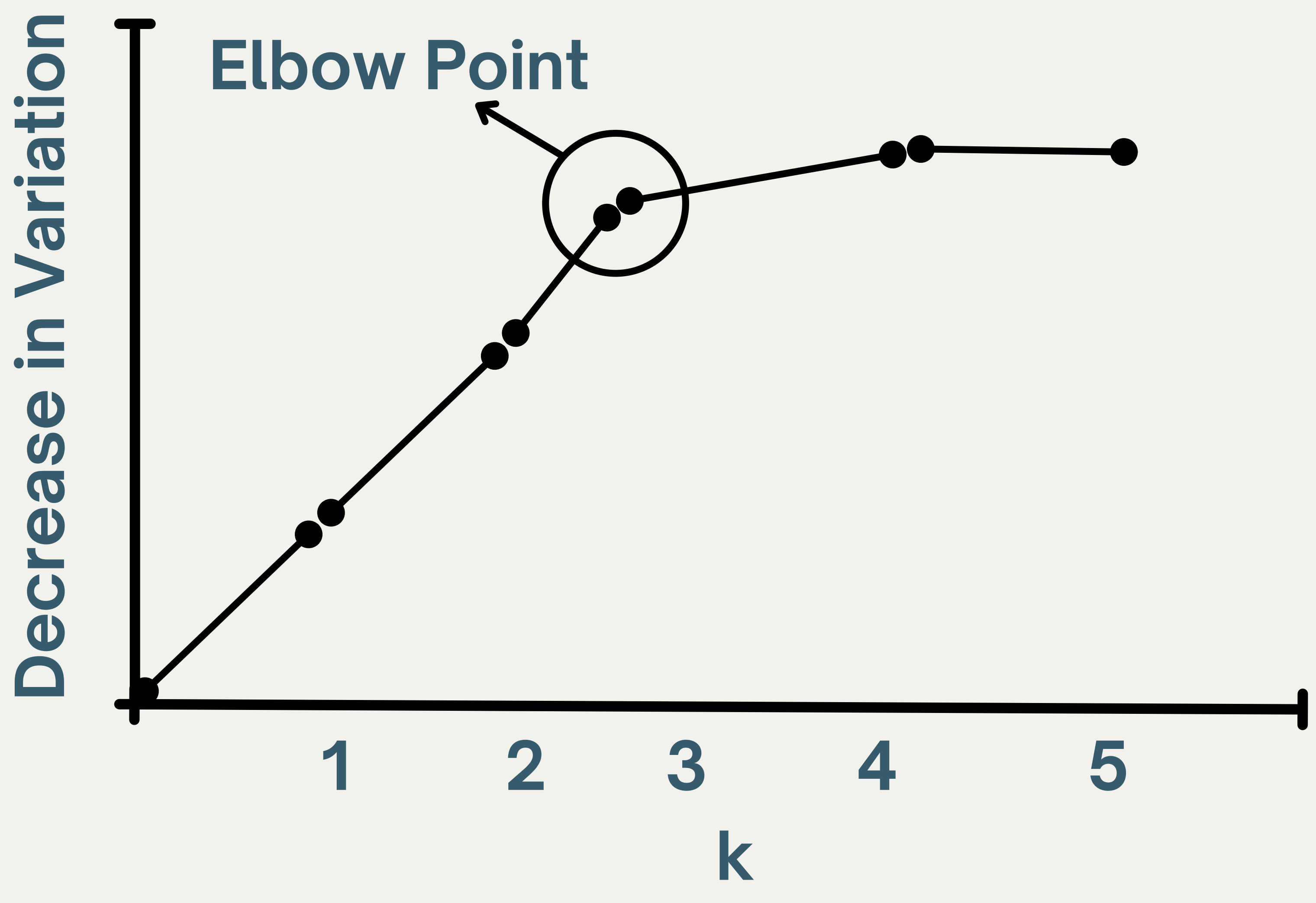
How do we decide K

A question people often have is, **how do we decide on the value of K?** In our examples it was somewhat clear by looking at the data, how many clusters would be ideal. But if it's a higher dimensional data, we will not be able to visualise it. How do we decide how many clusters should we create?

Since with each iteration of the centroids, the distance between the points and centroid decreases, we are essentially decreasing the **varations** within a cluster. **Variation** would simply be the sum of the square distances between the centroid and the data points.

$$\sum_{1}^{k} \sum ||x - c_k||^2$$

With each iteration, this varation will decrease. However, at some point, the decrease in variance will start to get smaller and smaller.



We usually choose a **K** that's at the **"Elbow Point"**, above which there's diminishing returns in creating more clusters.

Let’s Summarise

Let’s summarise what K-means Clustering is:

1. **What It Is**
    - A simple and popular unsupervised learning algorithm.
    - Groups data into K clusters based on similarity.

2. **How It Works**
    - Step 1: Choose K
    - Step 2: Initialize K centroids randomly.
    - Step 3: Assign data points to the nearest centroid.
    - Step 4: Update centroids as the mean of assigned points.
    - Step 5: Repeat until centroids stabilize.
3. **Goal**
    - Minimize within-cluster variance (make clusters compact).

4. **Strengths**
    - Easy to understand and implement.
    - Works well for structured, spherical data.

5. **Challenges**
    - Requires predefining K (number of clusters).
    - Sensitive to initialization and outliers.
    - Assumes clusters are spherical and similar in size.

6. **Applications**
    - Market Segmentation: Group customers by behavior.
    - Anomaly Detection: Identify unusual patterns.
    - Recommender Systems: Cluster similar users or items.
    - Image Compression: Reduce pixel redundancy.

Enjoyed reading?

Follow for everything Data and AI!

https://www.linkedin.com/company/code2career-ai